Assessment #4

Submission deadline: 15th May 2022 11:55pm AEST via Moodle

Instruction

Below are the coding tasks that you need to complete individually for assessment 4. You should download the IntelliJ project folder as below and unzip it. Then work on the tasks in the project folder.

Assessment4.zip

This assessment is worth 18% of the unit total. It contains 70 marks, which has two components.

- Task correctness weighted 60 marks.
  - Task 1 has 10 marks
  - Task 2 has 5 marks
  - Task 3 has 5 marks
  - Task 4 has 20 marks
  - Task 5 has 5 marks
  - Task 6 has 10 marks
  - Task 7 has 5 marks
- Code readability & documentation has 5 marks.
- Code development has 5 marks.

Academic Integrity

Please be reminded of the academic integrity mentioned in Week 01. You should code alone and ask the unit staff for help. Do not post your code in public forums.

Scenario

MONASH UNIVERSITY WOULD LIKE TO CREATE AN APP TO MANAGE UNIT (SUBJECT) OFFERED BY FACULTY IN EVERY SEMESTER. THIS APP WILL CONSIST THE DETAILS OF THE UNIT.

YOU ARE REQUIRED TO DESIGN A JAVA CLASS FOR THE APP BASED ON THE SPECIFICATIONS AS BELOW TASKS

Task 1 (W8 - 10 marks)

Code the class shell and instance variables for unit offered in a faculty. The class should be called `Unit`. A `Unit` instance has the following attributes:

- **unitCode**: length of 7 characters (you can assume all unit code is 7 characters in length)
- **unitName**: length of 40 characters max
- **creditHour**: represent by integer. Each unit has 6 credit hours by default unless specified.
- **offerFaculty**: length of 20 characters. It will be the name of the faculty that offer the unit (eg. Faculty of IT).
- **offeredThisSemester?**: Can be true or false (if true – offered this semester, if false – not offered)

Task 2 (W8 - 5 marks)

Code a non default three-parameter **constructor** with parameters for **unit code**, **credit hour** and **offer faculty**. Instance variables that are not taking parameter must be auto-initiliased with sensible default value. The constructor must utilise appropriate naming conventions and they protect the integrity of the class's instance variables.

Task 3 (W8 - 5 marks)

Code the **getter/accessor** methods for all the instance variables in task 1.

Task 4 (W8 - 20 marks)

Code the **setter/mutator methods** for all the instance variables in task 1and at least one of the method should return a boolean to indicate the success or failure of the mutation. The code must protect the integrity of the class's instance variables as required and utilise appropriate naming conventions.

Task 5 (W8 - 5 marks)

Code a **toString** method for the class that output as below.

```
Unit Code: FIT1051
Offered Faculty: Faculty of IT
Credit Hour: 6
Offered in this Semester? : true
```

Task 6 (W8 - 10 marks)

Code a two parameters method called **customCreditHour** that takes in a **unit code** and **number of credit hours.** This method should use the parameters and check if the first 3 character of the unit code is not **"FIT"**, then accept the unit code and number of credit hours. If the first 3 character is **"FIT"**, then it should print an error message ***"Error. This is FIT unit and the no of credit hours is 6 by default"***

e.g. calling the method **customCreditHour**("FIT1051", 12) will return ***"Error. This is FIT unit and the no of credit hours is 6 by default" .*** Declare your variables as appropriate.

You may assume any unit which is not offered by FIT should not be 6 credit hours by default.

Task 7 (W8 - 5 marks)

Write the main method code in a **UnitDriver** class that instantiates **Unit** objects to test successful and unsuccessful attempt as below.

```
=======Successful Attempt========
Unit Code: FIT1051
Offered Faculty: Faculty of IT
Credit Hour: 6
Offered in this Semester? : true

=======Unsuccessful Attempt=======
Unit Code: null
Offered Faculty: FIT
Credit Hour: 0
Offered in this Semester? : false
```

Code Readability (5 marks)

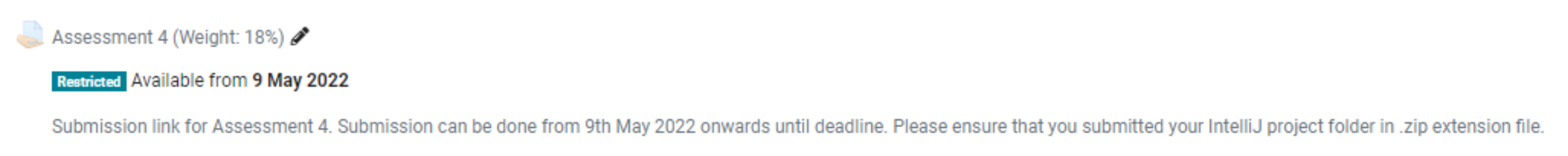
Overall code submission must be well organised and very easy to follow included but not limited to code indentation, code consistency, effective use of whitespace etc.

Code Development & Documentation (5 marks)

Overall code submission demonstrates correct method shell, syntax usage and meaningful naming conventions. Code documentations/inline comments are thorough and in detail.

Submission Instruction

Please submit your IntelliJ project folder as a .zip file and submit to via Moodle as below. If you are not sure how to zip your project, please refer to the video here.



Assessment 4 (Weight: 18%)

Restricted Available from **9 May 2022**

Submission link for Assessment 4. Submission can be done from 9th May 2022 onwards until deadline. Please ensure that you submitted your IntelliJ project folder in .zip extension file.

Marking Rubric

Assessment#4 Marking Rubric.pdf